1/2 Sub new

TS Bulletin #6-88

For Low Cost Computing
Pub. by
BILL HARMER
97 Ruskin St.,
Ottawa, Can. K1Y4B3

This issue in your hands offers useful programming tips and information as well as a pop-up subroutine to add to your ZX-81/TS1000 BASIC programs. Pop-up programs are quite a fad in home computing, so why not? This issue also marks a milestone with the publication of our 100th page of meaty article (ephemeral news and introductions are not counted in this). Below is a photo of some of our previous issues as the form of title and style of masthead changed. May we continue to provide this service into another year of TS activity as meetings for most groups resume in the fall after a summer break.

☞ Corrections: Contrary to what was said a few months ago, Knighted Computers, 10 Canalview Mall, Fulton, NY13069 is still supporting the TS2068. Catalog US$2....Larken Electronics advises that the mem. addresses in its disk drive manual for the ZX-81/TS1000 system (page 6) should be changed to: Cline, 12316,Sort 14480. This applies only to ZX-81/TS1000 systems. TS2068 system manual info is okay....Other News: Pascal anyone? Contact: D. Solly, 1545 Alta Vista Dr., Apt. 1402, Ottawa, Ont., Canada K1G 3P4 to join other users.

CONTENTS:

-Speeding Up BASIC Programs (Programming Tips). p.6
-Pop-Up Programs for the ZX-81/TS1000 . . . . p.2
-Preventing Computer Breakdown (Hardware Feature) p.5
-Motorola's New 88000 Microprocessor (News) p.4
-Six Facts You Might Not Know About the ZX-81/TS1000 . . p.7
-Writing for a Computer User Newsletter p.3
-Programming in ZX-81/TS1000 BASIC (Pros and Cons). . . . p.8

This is a public domain newsletter published in order to encourage the idea that you don't have to be well off to enjoy the home computing hobby. It is a non-profit project of the author/publisher alone and for that reason your support and tolerance are solicited for without it the newsletter would become more of a burden than a pleasure to do and would be killed. Copies are sent to a few user groups. Feel free to photocopy this as much as possible and distribute it among your friends. Due to the 'on the cheapest basis' philosophy of the publisher, subscriptions regretably are not available, although back issues are for about $2 each. (Ask first).

UTILITY FOR POP-UP FUNCTIONS (ZX-81/TS1000)

```
7000 REM POP-UP NOTE SUBROUTINE         (do not key in notes in small letters!)
7010 LET W$(1)="POP-UP NOTE UTILITY"
7020 LET W$(2)="ENTER A SHORT REMINDER MEMO"  (underlining means inverse)
7025 GOSUB 7100
7030 INPUT A$
7035 IF A$<>"" THEN LET W$(3)=A$    (if you just press enter message kept)
7040 PRINT AT 20,1;"REMINDER=bbbbbbbbbbbbbbbbbbbbb";AT 21,1; W$(3) (b=blank)
7050 LET W$(2)="PRESS ENTER TO GO ON----NOTE="
7055 PRINT AT 20,1;W$(2)
7060 INPUT A$
7070 GOSUB 7200
7090 RETURN
7100 REM POP-UP SUBROUTINE
7105 FAST
7110 LET SF=PEEK 16396+256*PEEK 16397  (This finds beginning of display file)
7115 LET C$=""
7120 FOR N=1 TO 4                                    the 33rd
7125 FOR X=1 TO 32            (This misses the line-end char./which is ENTER)
7130 LET C$=C$ + CHR$ (PEEK (SF + 594 + X +33 * (N-1)))
7140 NEXT X
7145 NEXT N
7150 SLOW
7155 PRINT AT 18,0;"┌(a whole line)──┐";"▐"; W$(1);"▌"; "▐"; W$(2); "▌";
     "▐"; W$(3); "▌"                          (Note: see text para3)
7199 RETURN
7200 REM POP/DOWN
7220 PRINT AT 18,0;C$
7299 RETURN
 10 DIM W$(3,30)    (place these lines where you dimension the other arrays in
 20 LET C$=""        your program)
 30 LET X=0     40 LET N=0    -------
```

Use of above:

These three subroutines will allow you to make pop-up sub-programs in your BASIC programs. GOSUB 7000 will call a note sub-program allowing a 30 character maximum length note to be inserted in string W$(2). If you wish to make your own pop-up program to say pop-up a simple calculator or formula evaluator or menu, you LET W$(1) be the first line of the pop-up display, W$(2) the 2nd and W$(3) the third. Then you GOSUB 7100 to pop it up and then continue with your sub-program. When you are ready to go back to the main program, GOSUB 7200 will restore the original screen bottom.

This program begs for improvement. It takes about 8 seconds for the pop-up screen to come up for example. The program is written with a view to easy understanding and entry from a listing and could be speeded up.(See below). You could also pop-up four lines rather than three. Change 594 to 561, DIM W$(3,30) to DIM W$(4,30) and so on.(N=1 TO 4 to N=1 TO 5). The PRINT AT numbers would have to be moved up too.

If the program halts at line 7010 with an error 2 then you have not dimensioned the array W$(x) and must run line 10 at the bottom of the listing. Line 7155 includes 30 graphic shifted "7"'s to make a border at top.

The purpose of having pop-up programs is to perform another function using your computer while the main program is 'on hold' so to speak, and can be resumed without stopping the program or having to rerun or reload it. A pop-up program differs from other ways of doing this in that the original screen display is restored when you return to the main program. Such programs are becoming very fashionable in the IBM-compatible world and our faithful ZX-81/TS1000 can do it too! (Note a faster routine make: 7120 LET SF=SF+594, 7125 FOR X=1 TO 132 STEP 3, 7130 LET C$=C$+CHR$(PEEK(SF+X))+CHR$(PEEK(SF+X+1))+CHR$(PEEK(SF+X+2)), 7145 LET C$=C$(1 TO 32)+C$(34 TO 65)+C$(67 TO 98)+C$(100 TO 131)→these changes speed the pop-up considerably).

6 Facts You Might Not Know About The ZX-81/TS1000

Here are some little-publicized facts about our Sinclair ZX-81/TS1000 computers:

Did you know that voice synthesizers, memory banks over 64K, EPROM cartridges, wordprocessors, data bases, spreadsheets, good keyboards, full-sized printer interfaces, disk drives, modem systems for Bulletin Board access, RS-232 interfaces, monochrome monitor interfaces, Centronic Interfaces, EPROM burners, and many other accessories have at one time or another been made for the ZX-81/TS1000? Some are still available new, others surplus or used.

Did you know that you can write operating system shells, assemblers, cross-translators/ cross-interpreters yourself without doing a bit of machine language programming? They can all be written in BASIC as can wordprocessors, spreadsheets and databases. The machine language programming of other, more expensive computers can also be partially simulated in programs for training or educational purposes using nothing but BASIC.

Did you know that the following languages were offered for the ZX-81/TS1000:

-Partial Pascal -a subset of Pascal, mainly integer
- LOGO, the children's language
-RPNZL, a language similar to FORTH, for the ZX-81/TS1000 only ?, RPN is Reverse Polish Notation, the L for Language
-FORTH, on cassette
-FORTH, in a ROM cartridge
-BASIC, compiler (integer)e.g. MCODER II
-several extensions of BASIC in the ZX-81/TS1000 to add READ and DATA statements, etc.

Did you know that if you are into machine language you can:

-enter machine language into a BASIC program using nothing more than a book as reference and a few tricks
-several assemblers have been made (they convert Z-80 assembly language into machine language). They include one by Artic, one by Gladstone and Hot-Z.
-machine language monitors that let you look at register contents and have other debuging aids were made
-disassemblers were made that took machine code, e.g. in the ROM and converted it to assembly language
-a training course of practice programs etc. that would run on the ZX-81 was made in Britain to teach machine language programming (Z-80)
-books of ROM disassembly have been written for the ZX-81
-books of routines in Z-80 machine language for doing arithmetic etc. have been written and you can borrow the routines from them for your own programs

Did you know that bugs in the machine language programs in your computer's ROM may give very wrong answers when calculations are done? Not just being a few decimal places out either, some bugs have been corrected in later ROMS and a replacement ROM was issued that claims to have caught them all. (Thos. Woods, not Sinclair or Timex sold this ROM).

Did you know that there is a trick problem in using BASIC programs longer than 16K in the ZX-81/TS1000 (up to the 32K maximum)? It can be done but you must use a trick.

Programming in ZX-81/TS1000 BASIC

Sinclair BASIC as implemented in the ZX-81 and TS1000 has some very fine features that are lacking in other home computers as well as some limitations. The first limitation that one notices is that chained lines joined with the colon (:) are not allowed. The TS2068 and most of the home computers of other makers (Commodore 64, IBM, Radio Shack Color Computer) allow this. The good news is that this makes the listings on the Sinclair machines easier to read and less crowded, making for easier debugging. The bad news is that a series of commands on separate lines will run a little more slowly than if placed all on one line in interpreted BASIC's (that is ROM BASIC's rather than compiled BASIC's). One problem that arises for ZX-81/TS1000 users converting programs written for other BASIC's is that an IF...THEN statement followed by a colon (:) and another statement will execute the statement after the colon only if the condition after the IF is true. Example:

```
   100   IF X=1 THEN PRINT "X IS ONE" : GOTO 200
```

must not be translated as:

```
100 IF X=1 THEN PRINT "X IS ONE"
105 GOTO 200
```

but can be translated as:

```
100 IF X=1 THEN PRINT "X IS ONE"
105 IF X=1 THEN GOTO 200
```

The most frustrating omission in ZX-81/TS1000 BASIC is the lack of READ and DATA statements. The use of these statements to contain machine code is not however much of a problem in translating programs from another computer to the ZX-81/TS1000 because such programs are usually untranslatable without rewriting the machine code routines from scratch. But READ and DATA statements are often used to do other things. If you read up on how these commands work and figure out what they do in the program it is not usually too hard to write a routine in ZX-81/TS1000 BASIC that will do the same thing. Using a GOSUB and a LET statment within an incremented loop using the IF...THEN GOTO has worked for me.

On the other hand there are some powerful advantages in ZX-81/TS1000 BASIC. The best is that most commands that will take a number will also take a variable name or a formula. This allows you to call a subroutine by name. It also allows you to simulate an operating system shell within a program. In most of my programs for example, the immediate command GOTO DISK will go to the routine that saves the program on (Larken System) disk. Similarly GOTO HELP can be used to go to the self-documenting or user instruction part of the program no matter which line it is stored at in various programs since you set up the line location with a LET HELP=9850 or whatever the line number it is in different programs. That way you don't have to remember a bunch of line numbers (possibly different ones) for each of your programs. And calling a subroutine or program module by a name such as CALCULATE or OUTPUT is a feature of many expensive languages that use that for a more structured approach to programming.

Too lots of other commands are better in ZX-81/TS1000 BASIC. To set up a title page in ZX-81/TS1000 dialect you can use CLS, PRINT AT, PAUSE, CLS and then move on. Try that with a VIC-20, Commodore 64 or TI-99. The Commodore 64 uses a subroutine with a ROM call to simulate PRINT AT, and the TI-99 uses a sort of ROM call. None of them has a PAUSE and the best you can do is substitute an empty FOR...NEXT loop to waste time. The clear screen is done on the Commodores by somewhat bizarre commands that are to make things harder not mentioned in the user's manual for each.

As we see most of the limitations of ZX-81/TS1000 can be easily gotten around and in addition it has several distinct advantages.